TABLE OF CONTENTS

TINY GEAP Programs

The Tiny GEAP programs included with your Dot Writer are designed to work independently of all other Dot Writer programs. They are our response to user requests for a simplified way to create HI-RES letters and graphics. The created result is fully compatible with all of our other programs. Tiny GEAP is simple to use and it is very fast. Read the instruction manual before operating and be sure that you operate ONLY WITH BACKUP DISKS!

WARRANTY

Tiny GEAP, like our other programs, is guaranteed to operate on the TRS-80 Model I and III. It is guaranteed to be fully compatible with TRSDOS. Although these programs have been tested with, and are currently compatible with most common Disk Operating Systems, we do not guarantee continued compatibility with any operating system other than TRSDOS.

Neither the author, nor RCM Computers, accepts liability for any damages that may result from the use or application of these programs. Our liability is limited to replacement of a faulty original program disk only. No other warranty is either expressed or implied.

Return Procedure

We will make every attempt to see that you have a working program. We do not, as a general rule, furnish refunds. We do not copy protect our programs and we therefore cannot refund opened or used programs. If you have located an incompatibility that makes these programs unusable to you, and if you are operating the programs on standard systems, we may refund the purchase price upon return of the programs, all documentation and a written statement that no copies have been made, retained or distributed. Please call or write before returning any programs. Be as specific about the incompatibility as you can and fully describe your system. The option to refund remains ours. We will refund under the most unusual circumstances only.

Copyright

All of our programs and documentation are copyrighted. We ask that you respect that copyright. Anyone caught violating our copyrights will be prosecuted under every applicable statute.

TINY GEAP Overview

TINY GEAP is a new program designed to ease the graphic and letterset creation task. We have listened to our users' input and we came to the following conclusions. Though GEAP is a fine program, it has much more power than the average user needs. That excess makes the use of GEAP more complicated than necessary. Our users suggested a minimum GEAP system that would allow simple creation of graphics and lettersets without all of the added GEAP features.

TINY GEAP to the rescue! We wrote TINY GEAP as a response to those user comments. TINY GEAP is much like GEAP however, it is very simple. There are 15 commands - most of which are used only once in any operation. We have also added an instruction menu that lists all of the commands. This menu can be displayed at any time without interfering with drawings on the screen.

TINY GEAP is fast. Since it is limited is scope and power, it was possible to increase speed. In fact, at the highest speed, the drawing cursor is too fast to use effectively.

Present users will have no trouble adapting to TINY GEAP and new users are in for a treat. Let's take a look at the operation of TINY GEAP:

1. TINY GEAP is used to create graphics that can be printed on the printer in HIGH RESOLUTION. The graphic can be a letter or a drawing taking up one or more screens.
2. The operation is simple. Once a working frame is defined (a simple matter of specifying the number of dots wide and high you want your graphic) you simply move the cursor around in the frame, leaving lines and dots where you desire. Once you have drawn your graphic, TINY GEAP will translate each pixel into a printer dot.
3. The resulting graphic can be dumped to the printer for a test, edited until it is just right, and then saved to a disk file for future use. In this fashion, an entire letterset can be created!
4. When the letterset is complete, it can be added to your collection and used with Dot Writer at any time!
5. We have also included a Transfer command that will allow you to load any letter already created onto the screen, and edit it. The result can be placed back into the original letterset or into a new one. This feature allows you to start with a STOCK letterset and create a new one.

Enough of this overview stuff. Let's take a look at the actual program and how it operates!

Disk File Names and Functions

In addition to the files mentioned in the Dot Writer manual, there are also three others. They are :

TGEAP1/EXT
TGEAP2/EXT
TGEAP3/EXT

The EXTension refers to the printer that the programs are compatible with. An "/ITO" extension will operate in the C.ITOH Prowriter 8510 or 8515, the PMC 8510 and the NEC 8023A. The "/EPS" extension stands for the EPSON and will work on every current version of the EPSON, including the new FX series. The only requirement is that the printer must have some version of GRAFTRAX.

ALL THREE OF THESE PROGRAMS MUST BE ON LINE IN ORDER TO RUN TINY GEAP. Other programs are optional and may be on the same diskette as the TGEAP programs but, and I reiterate - you must have at least the three TGEAP programs on line!

You will also note that there are programs with no extensions. These programs are printer independent - they will work on all printers. Those programs are explained in the Dot Writer manual. In this manual, we are concerned only with the three programs listed above.

Here is an explanation of the three programs. TGEAP1/EXT is a logo display and a

machine language loader. It serves two purposes, namely: display the LOGO as shown in figure 1, (Appendix A) and to load the machine language program, TGEAP2/EXT.

TGEAP2/EXT is the machine language module that is the heart of the graphics generation. It is entirely invisible to the user. It will be loaded automatically and will then be utilized by TGEAP3/EXT.

TGEAP3/EXT has many purposes. It is primarily a BASIC language program that calls and uses TGEAP2/EXT when speed is needed. TGEAP3/EXT is the brains and guts of TINY GEAP. It displays the command listing (as shown in figure 2 Appendix A) and it also formats, creates and translates the screen graphics into HI-RES.

Ever since we started marketing our programs, we have been asked, "WHY BASIC"? I'd like to explain the reasons why BASIC is important. First, we use BASIC only for those operations where speed is not important. Things like Disk I/O, module manipulation, etc. do not require great speed - BASIC can easily keep up. We use machine code where speed is needed and practical. In this fashion we accomplish several programming goals. We are able to easily modify the programs. We can easily alert users to changes that they might want to make. We can easily transport the programs from one computer to another and from one operating system to another. Also, machine code is larger than BASIC and we need to conserve memory.

The next reason is one I have already covered. Memory space. The TRS-80 is limited in memory size. Since we are dealing with large amounts of data, we want as much room as possible to be available. If we program entirely in machine language, we must do one of two things. We must use up tremendous memory space to rewrite routines that are already available in ROM or we must use ROM routines to conserve memory. If we write our own routines, duplicating ROM routines, we won't have enough space for all of our added features. This is especially true of Dot Writer! In the second case, we can save some space by using ROM routines but we sacrifice transportability since all computers do not use the same ROM. Trying to reach a happy medium is called "effective programming"!

Another important consideration is the ease of code manipulation. We want to support as many printers as possible. That means making changes to our code on a fairly regular basis. This is much easier in BASIC where we can document functions in the source code.

BASIC also makes it easier for the user to change the code if he desires. And, one last consideration, as we add new features, BASIC gives the flexibility of making the additions quickly and we can pass those additions on to the user for him to change, rather than charge for updates!

So, in the overall picture, using BASIC command modules really doesn't hinder the operation of the program. Look at it this way. We have very few competitors. Those that do put out similar programs have run into the same limitations we have. Some have beat those limits in the same way we did, others have fewer commands, less power, are limited to one or two disk operating systems and in general, gain only a very small speed margin. The speed increase is limited by the printer anyway. The EPSON is fairly slow in bit image mode and the TRS-80 BASIC is fast enough when aided by a few machine language modules. The faster printers can be spooled for greater processing speed but the increase is still not that great!

Running TINY GEAP

This is easy. First, make a BACKUP. Never work with the original disk. If you are like me and can't resist the temptation to mess around with the programs, a backup will save you a lot of grief. If you should blow up your diskette, return it with self addressed, stamped envelope and packing material and we will send you a new copy.

Now, working with the BACKUP disk. Boot your system. Depending on the Disk Operating System you are using, you will have some type of DOS READY prompt. Follow the instructions in your DOS manual for loading BASIC and be sure that you specify 4 files. In TRSDOS, you simply type BASIC <ENTER>. BASIC will load and you will be asked for: NUMBER OF FILES? - answer 4 <ENTER>. Don't worry about memory size. If you want to load a machine language routine in high

memory, be sure to protect it with a proper memory size, but TGEAP does not require any protection.

You should now be in BASIC with a READY PROMPT. Type RUN"TGEAP1/EXT (again, the EXT will be ITO for the C.Itoh compatible printers and EPS for the Epsons). If you have done everything correctly, you will see the logo on the screen. The logo will be displayed for a few seconds while TGEAP2/EXT and TGEAP3/EXT are loaded.

Finally, the logo will disappear and the screen will display the command summary that is shown in figure 2 (Appendix A). This command summary contains all you need to run the TINY GEAP programs. Note the "L" command. This command will display the command summary at any time without disrupting your screen work. Now let's examine the commands!

TINY GEAP Character Creation

Note that the last line of the command summary says "Hit <ENTER> to continue". Hit the ENTER key now. The screen will clear and you will have a flashing " - " (minus sign) in the upper right corner of the screen. This signifies the command mode. Whenever you are in this mode, you can access any of the commands on the command summary chart. Press the "L" key and watch as the command summary returns. Now hit the ENTER again and return to the command mode.

I'm going to briefly review each command in the order they appear. When the review is finished, we will run through a creation session. You'll be surprised at how simple the operation is.

The Save Command

First is the "S" - save figure to disk. When you have finished editing, creating, or manipulating a figure, you can save it back to disk with this command. You must be in the command mode (flashing minus) and if you are not, simply press the minus key.

There are several things you will need to know about this command. First, it saves whatever is within your screen frame to disk. Second, it saves it under a keyboard symbol. Third, each keyboard symbol has a special storage area on the disk.

The keyboard is arranged in ASCII order and so is the letterset disk file. the "!" is the lowest ASCII value and the "z" is the highest. There is a cheat sheet with an ASCII chart supplied with this manual for reference purposes. The "S" command saves in ASCII order so that if you save a graphic under the "z" character, the entire file will be set up and space reserved for all ASCII values below "z". Since "z" is the highest keyboard ASCII value, room for all other characters will be reserved.

On the other hand, if you save a graphic under the "!" character, no room is made below that character since it is the lowest ASCII value. The purpose of this information is this: If you are creating an entire letterset, it doesn't matter what order you save them in since you will always have enough space. If, on the other hand, you are creating only one or two graphics, it is best to save them under the lowest ASCII value keyboard characters so as to conserve disk space.

Now let's look at the "S"ave procedure. Like all of the commands, "S"ave is self prompting. Once you have completed a graphic, press the **minus key** to return to the command mode. The flashing minus will appear. Now press the "S" key. The screen will clear and you will be asked for the keyboard character under which you want to store the graphic. If it is a picture and only a few will be in the file, answer with a low ASCII value character. If you are creating a letterset, answer with the corresponding letter. An "A" for A or a "c" for c, etc. Numbers, letters, special characters, it doesn't matter what you store under any given keyboard character. The only convention to follow is to match the keyboard with the proper character when creating a letterset.

Once you have responded to the last prompt, you will be asked to enter a filename for the character. If you have been working on a file, that name will appear as the default, otherwise, you can enter any filename that is valid under you DOS. When you respond to the filename

prompt, the character will be saved and you will be returned to the command mode for your next function.

The Transfer Command

The next command on the menu is the "T"ransfer command. Again, you must be in the command mode (flashing minus) to enter any of these commands. If you are not in command mode, press the minus key.

The "T"ransfer command is designed to allow you to load a letter or graphic from a disk file for editing. Transfer automatically creates a frame for the loaded character so if you are working on creating a letterset, you do not have to remember the frame size. If you left off with the letter "J", just "T"ransfer that letter (or any other letter in the letterset) and the frame will be restored so that you can continue to create.

Here is the procedure. From the command mode (flashing minus), press the "T". You will be prompted for the keyboard character that you wish to enter. Answer with the letter that you want to load. You will then be asked for the filename of the file you want the letter to come from. Enter the filename. The letterset will then be opened, the letter will be loaded to the screen and the frame will be created. Now you can enter the graphic mode and edit or recreate the letter.

The Write Command

The next command is the "W"rite command. This command allows you to print, in high resolution, the character that is presently in the screen frame. Think of this as sort of a preview of the character.

Once the character is in the frame and you are ready for a test printout, simply go to the command mode (press the minus key) and press the "W" key. Be sure that the printer is on. The character that is in the frame will dump to the printer in hi-res!

The Assign Frame Command

Now we will look at the "A"ssign command. While in the command mode (flashing minus), hitting the "A" will allow you to "A"ssign a frame size within which to work. You will be prompted for the number of dots wide you want the frame. This will correspond directly to the number of individual pixels wide the frame will be. It will also be the number of printer dots wide the letter or graphic will be. You should carefully think this decision out. Experiment with different widths to find the best one for the letter or graphic you're drawing. There are some pointers that will help you decide on a frame size later on in this section. A single letterset can only contain one frame size!

The next prompt is for the number of printer lines. Each line is eight (8) dots high. If you select 1 printer line, you will have eight dots high to work in. If you select 2 you will have 16 dots high, etc. This is a printer limitation and not a program limitation. The printer only uses 8 of the nine dots in the vertical parameter and you must program all eight. Look in the following section on pointers for some more information on this point.

Frame Size Pointers

1. The screen size of the TRS-80 allows you to create a maximum frame size of 128 wide by 6 high. This is limiting in that you cannot create a single character larger than 128 by 6. The Letterset Manuiplation Utility program, which is available at reasonable cost, allows up to 16 screens to be used for one graphic. Letters larger than 128 by 6 will seldom be needed.

2. You can always make the graphic or letter smaller than the frame but never larger. If you are making a graphic drawing, you need only make the frame large enough to hold the graphic - it doesn't hurt to go too big.
3. If you are creating a letterset, you will want to use the smallest frame possible and unless you also have the Letterset Manipulation Utilities you cannot place the created letters into a smaller frame. In order to figure out how to select the width, we suggest that you try creating a large letter first. Try creating an "M" for width. The "M" should fill the frame. The other letters will then fit - width wise.
4. You might also want to consider the style of the letters. If you are creating a fairly common letterset, the "M" will be fine but if you plan to flare the letters, the "W" might be a better choice for the width selection.
5. Height is another problem. You will have to allow for descenders if you plan to have them. This can be done in several ways. Once you have selected a width, you must decide how much room you need to complete a descended character. Use a "g" or "q" for this purpose. Again, the style of the letter may need to be taken into consideration.
6. Remember, you must select height in increments of eight (8) so think ahead. If you have selected 1 printer line (8 dots) and you create your uppercase letters so that they touch the bottom of the frame, you will not get descenders that fall below the line of the uppercase. You must create your uppercase enough dots above the bottom of the frame to allow for the descenders! Eight dot high characters print faster than multi-line lettersets because they require only one pass. Our MicroPrint and Clarity are examples of the 8 dot high lettersets.
7. Finally, I mentioned the Letterset Manipulation Utilities once before. These utilities allow you to switch letters into different frames (among other things). If you have these utilities, you can make a frame larger than you need, create the letterset, then use the utility to find out the minimum frame size you can use for the letterset. The utility will then create the new frame size and allow you to move your letterset from the larger to the smaller frame. You can also move sections of the letterset so that you can mix and match lowercase and uppercase letters. I advise these utilities for anyone who is serious about making lettersets. They are very reasonably priced and well worth the money. There are 12 utilities and they are fully menu driven.

The Graphic Mode

Next is the "." (period) key. This command exits the command mode and enters the graphic mode. If you do not have a frame specified, you will be warned and asked to create a frame. If you have used the "F" command or the "T" the "." (period) will take you to the graphic creation page and you will see either the blank frame or the transfered character. While in this mode, you will see a flashing dot (single pixel) cursor within the frame. By pressing the minus key, the cursor will change to the flashing minus and hitting the "." (period) will bring back the flashing dot.

The Frame Redraw Command

The next command is the "F"rame draw command. You will almost never use it. It simply redraws your prespecified frame if you have erased it. The frame and its contents, can be erased by simply pressing the CLEAR key while you are in the graphic (flashing dot) mode. The entire screen will clear and you will be returned to the command mode (flashing minus). Pressing the "F" will redraw your frame so that you can continue to create letters.

Determining Storage Size

The "H"ow big command is uses to determine storage size of a letterset. Once a frame has been created, simply go to the command mode and press the "H". The frame size will be evaluated and a display will show you how large your letterset will be. The maximum letterset size is 32767 bytes and some frame sizes will exceed this limit. That is the reason for the "H" command. If you

were to specify a frame of 128 wide by 6 high, the "H" command would return the following display:

```
Memory Requirements For Current Letters:
! to /   13056 Bytes
! to 9   19968 Bytes
! to Z   44544 Bytes
! to z   73728 Bytes
Maximum File length is 32767 Bytes
Hit <ENTER> to continue?
```

As you can see, the entire letterset will not fit in the disk file. It will also not fit into memory all at one time! It is up to you to decide how many and which characters you will use in this case. Most graphics will not give you any trouble and most lettersets will be small enough to fit into the disk file. If they don't, you must decide which ones you want. Use the "H"ow big command to decide where to quit. Just a note: you can't beat this problem by placing the first half of the alphabet in one file and the second in another. While the theory is good, the second half of the alphabet would still have to be stored under the first half keyboard characters since the file size is determined by the highest ASCII value character.

The Command Summary Display "L"

You have already seen the "L"ist command is action. While in the command mode (flashing minus) and when the frame is displayed, typing the "L" will give you a display as in figure 2 (Appendix A). This is a command summary. The character or graphic will not be disturbed and you can press the "." (period) key to return to it.

The Exit Options "E"

The "E" command is simple to use. While in the command mode (flashing minus), pressing the "E" will exit to a menu which gives you four options. They are as follows:

1. GEAP MENU - will run the Graphics Editor And Programmer menu ONLY IF IT IS ON LINE. If you don't have GEAP or don't have it on line, an error will occur and you will be returned to the command mode.
2. NEWSCRIPT - will run NewScript if it is on line. If not, the same error as above will occur. In order for this to work properly, you must have had NewScript loaded and run BEFORE running TGEAP1/EXT. NS/CMD must be in memory and initialized.
3. BASIC - dumps command to DOS BASIC.
4. CANCEL - returns you to the TGEAP program so that you can continue with your work.

Using The Arrows

Finally, you can move the flashing minus sign by using the arrows. They can be used alone or in combination. With TGEAP, there is no real need for this ability but it is there in case of future enhancements..

Graphic Mode Commands

Now that you have seen how the command mode works, let's take a look at the graphic commands. This is very simple so I'll just run through them in a summary format:

1. Using the arrows will move the cursor (in this case the flashing dot) in the indicated direction. The cursor will stay within the frame and will "wrap around" when frame boundries are encountered.

2. Holding the <SHIFT> key while moving the cursor will cause a line to be left behind. This is how we draw the characters. The dot cursor is destructive so you cannot move it over a character without holding the <SHIFT> key. If you want to move around a large frame, go to the command mode (flashing minus) and move the cursor. It is not destructive.
3. While the flashing dot is present, hitting the <CLEAR> key will clear the ENTIRE screen. You can regain the frame by pressing the "F" key while in the command mode, but you will lose the graphic you were working on.
4. Pressing the minus sign, will return you to the command mode (flashing minus cursor). You will have to due this in order to use any of the command mode commands!
5. Finally, cursor speed is controlled by the numbers 0 - 7. The higher the number, the slower the cursor. Number 7 will cause the cursor to creep along and 0 is so fast that is is almost unusable.

That concludes the command summary and instruction. Most of you will be able to run the program now and will create letters without any difficulty. For those who are still a bit cloudy, the following section will walk you through a letter creation.

The Walk Through Example

1. Run the TGEAP1/EXT program for your printer as explained earlier in the text. When the command summary is displayed, hit the <ENTER> key. You will now have a blank screen with a flashing minus in the upper left corner of the screen.
2. For our example, we will need to create a frame, a letter and a file to play with. For now, press the "." (period) key. See the error message? That will happen whenever there is an attempt to do something that requires a frame and a frame has not yet been created. A frame can be created with the "A" command or an old frame can be loaded with the "T" command. Look back to refresh your memory on how these operate.
3. Let's create a frame. Press the "A". You will be asked for the number of dots wide. Use 15 for now. Next you will be asked for the number of printer lines high. Remember these are in a group of eight dots for each printer line. Select 2 for now.
4. After you have answered the questions, the screen will clear, a frame will appear in the upper left corner and you will be back in the command mode. Now press the "." (period) key. The cursor changed to a dot. Use the ARROWS ONLY to move it around the frame. Also, try the 0-7 keys and watch how the speed is changed. Also note that the cursor won't leave the frame.
5. Now, hit the <CLEAR> key. The screen is cleared and the command mode cursor (flashing minus) is back. Anytime you want to clear the screen, this procedure will work. If you have specified a frame size, change your mind and specify a new frame size, both frames will be on the screen. Using the <CLEAR> will clear the screen and the "F" will restore the current (newest) frame. You can try creating multiple frames now but be sure that the last one you use is the 15 wide by 2 high. That way you will be in step with our example.
6. The screen is now clear and the flashing minus is on the screen. Press "F". The frame reappears. It's time to draw a letter. Make it simple. The letter "L" is nice for this. Remember that you will have to stay inside the frame and that you will be able to adjust the speed of the cursor. Press the "." (period) key and use the arrows and <SHIFT> to draw the lines.
7. You should now have something on the screen. It doesn't matter what. Press the minus key and watch the cursor change. You are now in the command mode. Be sure that your printer is on and press the "W" key. The character in the frame will print out. The result is how the character will always look. The screen will restore to normal and you can edit the character until it looks right. Play with it a while and see how it works.
8. I hope that by now you have created a character that you are satisfied with. Let's see how many of these characters you can store to disk. Check the command summary by pressing the minus key (enter command mode) and then press the "L" key. The commands will list. See the one we want? It is the "H" command. Press <ENTER> and go back to the screen. You will still be in the command mode. Press "H". The following information will display:

! to / 510 Bytes
! to 9 700 Bytes
! to Z 1740 Bytes
! to z 2880 Bytes

We have plenty of room. Actually, you only need to check this when you are working with large frames.

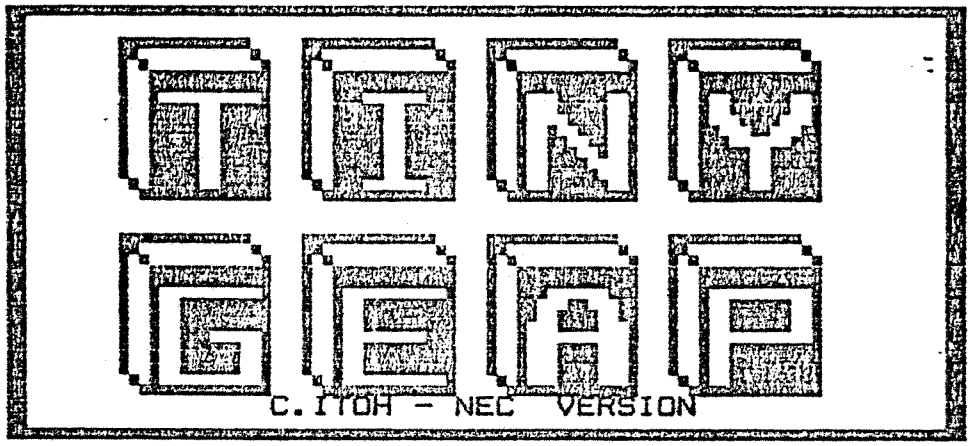
9. Now hit <ENTER> and return to your character. Let's save the little bugger to the disk. Rather than save it under "L" or whatever letter you drew, we'll save it under the lowest ASCII value to save disk space. You should be in the command mode (flashing minus), if not, get there. Now press the "S" key for save. You will be prompted for the keyboard character you want to save your graphic under. Answer "!", without the quotes of course. This is the lowest ASCII value and will not reserve any disk space below it. The next prompt is for the filename. Use and valid file name. Try : TEST/LET.PASS:1. This will save the letter under the file name TEST with the extension LET and the password PASS. It will also put the file on the disk in drive 1. Feel free to alter this name in any way you want.
10. OK! That is it. You have created and saved a letter. Let's try to load it back in now. For the sake of experimentation, reboot your computer. Start from scratch. When you have the command menu displayed, hit <ENTER> and go to a blank screen. You will be in the command (flashing minus) mode. Press "F" - nothing happens. Press "." (period) and you get an error message. There is no frame defined!
11. Now press the "T" key. You will be asked for the keyboard character that you want to load. Type "!" (again, without the quotes). You will be prompted for the filename. Enter the filename that you used to save your character. The access to disk is made and - surprise - the character appears on the screen and in the frame.
12. Just to be sure that you have the frame there, enter the graphic mode (press the period) and then clear the screen (press the <CLEAR>). Now press the "F" and you will see your frame reappear. No letter - we erased it.
13. Type "T" now and specify the "!" to the prompt. Note that the second prompt now has a default. It is the filename that you entered a few minutes ago. Press <ENTER> only and the letter will reload and redisplay.

That is it! It is that simple. If you want some fun, try loading in some of the characters from our supplied fonts. Look at them to see how they were made. Alter them and print them out (use "W") and see how they look. Remember not to permanently change an original font style. You might regret it. You now have mastered Tiny GEAP!

For information on using the lettersets you create, see the DOTPRINT portion of the manual. For further information on manipulation of the fonts, see the Letterset Manipulation Utility manual.

Appendix A - Figures 1 and 2

FIG. 1



(C) 1983 BY
WILLIAM K. MASON

FIG. 2

COMMANDS WHEN FLASHING - IS PRESENT

S save figure to disk	F draw frame
T transfer figure from disk	H display storage limits
W write screen to printer	L list instructions
A set dimensions for figure	E exit
. go to flashing ▫	Arrow(s) move flashing -

COMMANDS WHEN FLASHING ▫ IS PRESENT

Arrow(s) Move flashing ▫ in indicated direction. Erases as it moves.
<SHIFT>+Arrow(s) Draw line in indicated direction.
<CLEAR> Clear screen.
- Return to flashing - sign.
0-7 Number keys control speed of flashing ▫.

(Hit <ENTER> to continue)

Appendix B - The ASCII Value Chart

VALUE	CHARACTER	VALUE	CHARACTER	VALUE	CHARACTER
32	SPACE	62	>	92	NA
33	!	63	?	93	NA
34	"	64	at sign	94	NA
35	#	65	A	95	NA
36	$	66	B	96	NA
37	%	67	C	97	a
38	&	68	D	98	b
39	'	69	E	99	c
40	(	70	F	100	d
41	)	71	G	101	e
42	*	72	H	102	f
43	+	73	I	103	g
44	,	74	J	104	h
45	-	75	K	105	i
46	.	76	L	106	j
47	/	77	M	107	k
48	0	78	N	108	l
49	1	79	O	109	m
50	2	80	P	110	n
51	3	81	Q	111	o
52	4	82	R	112	p
53	5	83	S	113	q
54	6	84	T	114	r
55	7	85	U	115	s
56	8	86	V	116	t
57	9	87	W	117	u
58	:	88	X	118	v
59	;	89	Y	119	w
60	<	90	Z	120	x
61	=	91	[	121	y
				122	z